Why Companies Should Develop Event Models

David Luckham
W. Roy Schulte

Introduction

Companies inherently depend upon events that are received from widely differing sources to control many aspects of their operations. They expect these operations to run in real-time. But in fact the processes that use events in the business decision making of most companies are not as automated or real time as they should be. At some stage a company (or more likely some person within the company) will realize that it could organize its business operations to take better advantage of the events that are flowing through the company.

When this happens, a process takes place in which the company gradually adopts event processing thinking and technology and becomes “event-driven”. It is usually a slow process of understanding and modifying existing information infrastructure. Personnel have to be educated in event driven methods.

This paper explains why a company should develop an event model to support its event-driven business applications. It is based on observations of current practices in several different types of business. The paper is informal and does not describe formal methods of event modeling that could be employed in the future.

Adopting Event-Driven Analytics

An event is something that happens. An event may be as trivial as a customer downloading a Web page or a refrigerator reporting its current temperature. Or it may be more substantial, such as bank payment, a change in a competitor's price, or a customer placing an order. Or as important as acquiring another company, or shutting down a factory for maintenance.

There are many real time sources of events. Data about events are constantly generated by business applications, email and other support systems within the business, low level sources such as RFID readers, and systems outside the business such as social computing apps, Web news feeds, industry associations and a myriad of other sources. The incoming events from these sources are sometimes called *base events* or *raw events.*

However all the data about events are rarely used to make business decisions in real time. Instead, much of the event data is put into data warehouses and other databases to be analyzed offline using traditional periodic business intelligence (BI) reports or predictive analytics tools. Only a fraction of the available event data is used in making real-time business decisions.

When a company reaches the stage of deciding to adopt event processing methods, its first decision is often whether to buy or build a real-time operational intelligence system ─ one that can provide current information about what is happening in the company to aid in the company’s business decisions. Most of these systems have dashboards or

displays on mobile devices that report recent events. More advanced systems provide key performance indicators (KPIs) derived from current events to give managers better visibility into their company's operations and its environment.

The adoption of real time event processing can be complicated and lengthy:

- Analysts are hired to study the business,
- Event-based analytical applications are designed,
- Event processing software to implement the applications is either purchased or developed,
- Data adaptors for connecting the event data sources to the analytics are installed,
- The analytical applications are deployed,
- Business processes are changed, and
- Users are educated in event-driven methods.

These steps are not necessarily carried out in this order – for example, users should be educated in event-driven methods before the analytical applications are deployed, and in some cases, may be educated when the event-based analytical applications are designed. All of the steps in this adoption process will be aided by the existence of a document that precisely defines the types of events that the company expects to use. This is called *an event model*. It acts as a reference guide. In fact, if an event model does not already exist it will probably need to be drawn up by analysts and agreed to by the company's business managers before or during the adoption process.

An event model for a large multi-faceted business, for example in transportation, manufacturing, retailing, or energy distribution, could run to thousands of different types of events. It is therefore a document that will evolve incrementally over a period of years. The steps of developing an event model for a company's business operations will be done one event driven application at a time, as needed. The effort involved is wrapped into the effort of application design and development. The interactions between events in different applications will lead to *sophisticated* event models (described below), and eventually to an event model covering much of the company's business.

Once in use, event-driven analytics and event models are continuously modified as business conditions change and users think of improvements. Additional types of events are brought into the picture and more users gain real-time access to events. Eventually many more aspects of the company come to leverage events in real-time.

What is an Event Model

An event is something that happened. An event is represented in computer systems by an object that contains data about the event, such as the time it happened, the location, what other events caused it to happen, and many other attributes. That object can be in the form of a message, rows in a database, records with multiple components or other types of complex objects. Such an object is also called "an event". So the word "event" has a double meaning, both as something that happened and as an object that represents that happening.

In this article the context of each use of the word “event” will determine whether the happening or the object is meant. Usually in what follows we mean the “event object”.

Event models describe events and event data. In some cases, a company needs only a basic model, while other cases call for a more-sophisticated model.

Basic event models include:

- A list of event types. When there are large numbers of event types, they may be organized into event taxonomies.
- Event schemas that specify the individual attributes (fields) in the events.
- Metadata on the sources and uses of events, conditions under which events are generated, event life cycle (how long with the event data be retained) and other characteristics.

Sophisticated event models will include (in addition to a basic model):

- A specification of the timing and causal relationships among events. This may be in text or in a diagram of an abstraction hierarchy that categorizes the events into levels.
- Detailed documentation on the logic (rules and algorithms) required to compute complex events from simpler base events.

Implicit in a sophisticated event model is the idea that not all events are equal, but that indeed some events may result from sets of base events and contain abstractions of the data in those base events. We will say more about *complex events* and the role of *event hierarchies* in event driven systems later.

Current practice

The use of event models is a relatively new idea. But it has analogies in business process management (BPM) and data management. Business processes, for example, always have underlying process models. In some cases, they are explicitly documented, but more often process models are simply “understood” and generally agreed to within a company. Similarly data objects always have implicit or explicit data models. Some business analysts and solution architects are systematic and explicit about how they develop and document process models and data models. However, companies do not invest the time needed to develop process or data models for simple applications.

Similarly few event processing projects develop and document a separate, basic event model, and very few develop a sophisticated event model. In most cases, there are only rudimentary informal documents (e.g., email) describing some of the events and event processing logic.

We note that a company’s event processing applications may be implemented in a variety of ways, e.g., in Java, a BI tool, a business activity monitoring (BAM) platform, a real-time operational intelligence platform, or an event processing platform. These tools need a description of the event data, including all of the attributes, and the rules that will be used to compute on the event data. So a de facto event model is embedded in the

software that performs the event processing. In some cases process or data models may include some data about events as a side issue, but common process and data models aren't equipped to represent all of the issues involved in a sophisticated event model, such as event timing and causal relationships.

In summary, the next step of formally documenting an event model separately from the application is rarely taken – but it should be. We believe that events should be first class citizens in a company's information architecture.

Reasons to Design Explicit Event Models

In most cases, development teams will benefit from explicit event models for the same reasons they benefit from explicit process and data models:

- The process of developing an event model is a helpful discipline for getting business analysts, solution architects, business decision makers and other subject matter experts to discuss the business problem and its solution.

- Once the event model is explicitly documented, the developers can implement the application more quickly.

- After the application is in use, the model helps developers find and fix bugs more easily.

- It also makes it easier to modify or extend the application because developers can understand more quickly the event processing that the system does.

- An explicit event model facilitates re-use of the events so that developers building new applications don't waste time and effort re-implementing the same or similar event detection capabilities that were already developed for earlier applications.

Basic event models help in systems that leverage multiple kinds of event data from multiple sources because they provide the metadata that developers use to maintain, modify and reuse the event data more easily. They are especially appropriate if event data is shared among multiple teams in different locations, or if the system lives long enough so that new generations of developers are brought on board to maintain and modify the system. Basic event models are relevant for analytical kinds of applications (the primary focus of this article) and for transactional application systems. Transactional systems typically don't use events to compute complex events or KPIs, but they often use events to trigger execution of some function.

Sophisticated event models are applicable if the system generates complex events rather than just using the basic events. This applies mostly to analytical event-processing applications, such as BAM applications. Sophisticated event models are especially valuable where complex events are based on detecting patterns of basic events rather than if the complex events result from simple aggregate computation (aggregates include functions such as sum or average). The event model helps users, analysts and developers visualize the relationships among the events and develop new patterns.

The value of event models in large, complicated and long-lived systems is becoming clearer, just as the value of process and data models became clear when transactional systems grew larger and more complicated.

However explicit documentation of an event model is an unnecessary burden for small systems, particularly those with only a few event types, those that don't compute complex events from simple events, or those that will only be used for a few weeks and then discarded.

The Process for Designing Event Models

Step 1: Understand the business

Business analysts begin by interviewing managers, first level supervisors, and individual contributors who make business decisions. They first need to understand the business, its business processes, what decisions are being made, which of the decisions need to be improved and what kind of data is being used. They may also collect information from operations analysts and other subject matter experts who can help identify how the business works.

This is the first step that occurs in any development project, and it should produce a process model, a data model and a decision model. The decision model documents who makes the business decisions, the decision factors, the decision process and the KPIs and other metrics that are used to make decisions.

A detailed decision model also specifies the business rules, goals and constraints so that a custom program or decision management software tool can produce prescriptive advice on what to do. The nature of a decision model depends on the kind of decision management software that is used. For example, if a rule engine is used, the decision model might include a business rule such as,

> "If a customer who calls into the company's contact center has spent more than $300 in the past year (and thus is "high value") is made to wait on hold for more than three minutes or is transferred more than twice, then have an agent make a follow-up call to them with an apology and a coupon for a free sample."

Other kinds of decision management software, such as statistically based scoring services or optimization or simulation tools, use other kinds of decision models.

Analysts investigate the kinds of information that are currently being used to make decisions to see if

(i) Additional metrics are needed or

(ii) Currently used metrics would be more valuable if they were more recent (fresher) or

(iii) Current metrics are sufficient but some additional computation should be provided to offload work from the decision maker

(iv) The metrics and alerts should be delivered through other channels (to a mobile device or through an email alert rather than simply displayed on a dashboard, for example)

Analysts will sometimes refer to the organization chart, because the organization hierarchy may influence the design of the event hierarchy. For example, a branch office manager may want to see a roll up of the sales activity in his or her branch, whereas a regional manager may want to see the total of the sales activities in the branches within the region. This familiar hierarchical structure is often used in traditional BI reports and may also apply to real-time analytic applications.

Step 2: Identify the complex events

The next step is to identify the complex events that will be needed to support the business decisions. A KPI or alert used in a decision model generally maps to one complex event. For example, the supervisor in a customer contact center may want to see the average caller wait time (a KPI) to make staffing assignments. This KPI is mapped to a complex-event type “wait_time_service_level” that has two attributes, “time_period” and “average_wait_time.” The complex event will be computed continuously and displayed as a KPI in a bar chart on a business dashboard that is refreshed every five minutes.

Step 3: Determine the base events and the event processing logic

Analysts work backwards from the complex-event description to identify the input base events and event processing logic that will be needed to generate those complex events. In our example, the base events are the records from all of the customer calls that occur within the five minute time window. Average_wait_time is the total of the wait_times for all phone calls during the most-recent five minutes divided by the number of calls. Event models are designed top down, working backwards from complex events to base events, so that analysts don’t have to document all of the base events that are potentially available. It would be impractical to start bottom up by documenting the thousands or more of the input event types that could potentially be collected from sources inside or outside of the company. An event model only needs to include the complex and base events that will be used to make decisions.

Step 4: Iterate as necessary

It is sometimes impossible to get the base events needed to produce the desired complex events. The analyst may have to go back to the decision maker to find alternative KPIs that could help the decision (for example, finding proxies for the metrics that the user really wanted) and repeat steps 2 and 3.

Step 5: Document the basic event model

5.1 Document the event types

The names of the complex and simple event types should be documented in a list or event taxonomy.

5.2 Document the event schemas

The attributes of the event types should be documented in a file, spreadsheet or (for a large, formally-managed IT program) in a metadata repository.

Step 6: Document the sophisticated event model

6.1 In a sophisticated event model, the relationships among simple and complex events should be documented in text or in abstraction hierarchy diagrams. Further explanation of complex events is included in *"Event Processing: Designing IT Systems for Agile Companies"* by K. Mani Chandy and W. Roy Schulte.

6.2 The rules and algorithms required to compute complex events from simpler "base" events should be documented in a causal model. Techniques for building causal models and abstraction hierarchies are presented in "*The Power of Events: an Introduction to Complex Event Processing in Distributed Enterprise Systems*," by Dr. David Luckham.

Additional explanation of the process for developing a sophisticated event model with examples will be presented in a future article.

Epilog:

Once the event model has been prepared, software developers can proceed with developing the application and the connections to the event sources. Analysts and users may be involved with tailoring the system, modifying event queries, adjusting parameters, and designing the look and feel of the dashboard (if there is one) and alerts (if there are any). Event processing applications are journeys, not destinations, so companies should expect the system to continue to evolve over time as business conditions changes and as users learn better ways of workings.

References

"Event Processing: Designing IT Systems for Agile Companies." K. Mani Chandy and W. Roy Schulte, McGraw-Hill, 2010, ISBN: 978-0-07-163350-5

"The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems." David Luckham, Addison-Wesley Professional, 2002, ISBN: 0-201-72789-7

"Event Processing for Business: Organizing the Real-time Enterprise." David Luckham, Wiley, 2011, ISBN: 978-0-470-53485-4.